A Design Workbench for Interactive Music Systems

Joseph Malloch, Jérémie Garcia, Marcelo M. Wanderley, Wendy Mackay, Michel Beaudouin-Lafon, Stéphane Huot

Author's version. Chapter in: S. Holland, T. Mudd, K. Wilkie-McKenna, A. McPherson & M. Wanderley, eds. New Directions in Music and Human-Computer Interaction. Springer. 2019. https://link.springer.com/chapter/10.1007/978-3-319-92069-6_2

A Design Workbench for Interactive Music Systems

Joseph Malloch[1], Jérémie Garcia[2], Marcelo M. Wanderley[3,6], Wendy E. Mackay[4], Michel Beaudouin-Lafon[5], and Stéphane Huot[6]

Abstract This chapter discusses possible links between the fields of computer music and human-computer interaction (HCI), particularly in the context of the MIDWAY project between Inria, France and McGill University, Canada. The goal of MIDWAY is to construct a "musical interaction design workbench" to facilitate the exploration and development of new interactive technologies for musical creation and performance by bringing together useful models, tools, and recent developments from computer music and HCI. Such models and tools can expand the means available for musical expression, as well as provide HCI researchers with a better foundation for the design of tools for "extreme" users. We conclude with a discussion of design guidelines for Interactive Music Systems.

1 Introduction

Since the appearance of digital tools, composers, musicians and designers have been inventing and crafting digital musical interfaces and interactions as a means to produce new sounds and to explore musical content. The designing of new interactive devices and tools intended for "musical expression" is particularly challenging, due partly to the idiosyncratic approaches and practices of our "users"—who often actively seek out ways to reimagine and recontextualise the application of standard tools—as well as to the lack of easily identifiable and quantifiable goals (Wanderley and Orio 2002). The broader field of human-computer interaction (HCI) often focuses on using interactive technologies to improve human performance, measured in terms of efficiency and accuracy; in contrast, music creation instead values concepts such as creativity, engagement, personalization, and appropriation. Evaluation

[1] Dalhousie University, Canada, joseph.malloch@dal.ca
[2] Université de Toulouse - ENAC, France, jeremie.garcia@enac.fr
[3] McGill University, Canada; Inria Lille - Nord Europe, France, mwanderley@acm.org
[4] Inria; LRI, Université Paris-Sud, CNRS; Université Paris-Saclay Orsay, France, mackay@lri.fr
[5] LRI, Université Paris-Sud, CNRS; Inria; Université Paris-Saclay Orsay, France, mbl@lri.fr
[6] Inria Lille - Nord Europe, France, stephane.huot@inria.fr

of new instruments and their use tends to be personal, subjective, instrument-specific (and even composition-, performance- or venue-specific) and difficult to generalize into standards or recommendations for informing future designs.

Our goal is to make the design of computer music systems less ad hoc, or at least to explore this possibility. We believe that the creative context of music provides opportunities for putting cutting-edge HCI models and tools into practice, and that there is strong potential for computer music technology that supports and embeds existing design models and methodologies (Huot 2013). New tools built from a well-defined design space will, in turn, facilitate validation and evaluation, exploration and extrapolation; and support reuse and appropriation for other fields. As a complementary problem for HCI researchers, expert musicians push the boundaries of system design through their personalization and appropriation of music applications, rendering standard HCI performance measures insufficient for evaluating musical tools. A more systematic application of HCI models to musical design and control will also highlight areas in which the models can be adapted, extended, or replaced.

In this chapter we argue that musicians, designers and researchers would all benefit from a "musical interaction workbench" comprised of relevant models and tools. We do not have space for an exhaustive accounting of models, tools and techniques, but rather attempt to sketch the foundations of an initial workbench that can be extended, enhanced and adapted to various contexts and needs.

This chapter first outlines key challenges for computer music and HCI research. It then articulates the need for a musical interaction workbench and presents key components of such a workbench in the context of previous work by ourselves and the broader research community. Finally, we build upon the workbench to formulate design guidelines and discuss future directions for the MIDWAY workbench.

1.1 Computer Music & HCI

Computer music and HCI have similarities and differences. To start with, both fields deal with ways to interact with computers. The specifics of these interactions, however, might differ substantially (cf. section 2 Models, below).

Hunt and Kirk presented an in-depth review of the differences between computer interfaces (mostly WIMP[7] based) and the interfaces of musical instruments:

> In stark contrast to the commonly accepted choice-based nature of many computer interfaces are the control interfaces for musical instruments and vehicles, where the human operator is totally in charge of the action. Many parameters are controlled simultaneously and the human operator has an overall view of what the system is doing. Feedback is gained not by on-screen prompts, but by experiencing the moment-by-moment effect of each action with the whole body. (2000)

[7] Windows, Icons, Menu, Pointer

This quote summarizes one of the key differences between these two contexts: the juxtaposition of *punctuated, dialogue-based interaction* where choices are made to answer requests from a partner and interactions based on a *continuous, bi-directional flow of information*. Although dialogue-based interactions are also found in computer music—for instance, in the case of live-coding systems—they are not the dominant way for interacting with computers in a musical context. Interaction in computer music can take many forms, ranging from performers playing digital musical instruments (DMIs) with goals and actions similar to performers of acoustic musical instruments (Miranda and Wanderley 2006), to live-coding or interactive music systems that are only distantly related to traditional musical instrument performances (Fig. 1).

Hunt and Kirk also suggest several useful attributes of instrumental real-time control systems helpful for differentiating them from classic HCI interactions, among them: *"There is no fixed ordering to the human-computer dialogue; There is no single permitted set of options (e.g. choices from a menu) but rather a series of continuous controls; Similar movements produce similar results; The overall control of the system [...] is the main goal, rather than the ordered transfer of information; Further practice develops increased control intimacy and thus competence of operation."* (ibid.)

Furthermore, there are several other major differences between computer music systems and WIMP/common HCI interfaces:

- *Timing is of the essence*. It is clear that time is an essential parameter in computer music interactions due to the inherent role of time in music. This often entails very stringent system requirements: usually low, constant latency and high temporal resolution, accuracy and repeatability (Medeiros and Wanderley 2014). The exact requirements vary depending on the desired type of control (cf. section 2 Models).
- *There is no easily definable task to evaluate*. In contrast to many interactions in HCI (e.g. pointing, text editing, navigation), the "task" in a musical performance is a dynamically-evolving blend of goals that include technical, aesthetic and cultural facets. It is not always easy to isolate variables of interest for performing quantitative evaluations of interfaces or instruments. It is difficult to judge whether a performance or an instrument is successful—even experts often disagree on what makes a successful musical performance! Methods for studying composition tools include collecting questionnaire data, interviews, and field studies with open-ended explorations of interactive tools (Eaglestone and Ford 2001; Fiebrink et al. 2010). These methods are often conducted with composers in the field to explore a particular use or approach. However, they are not suitable for comparing how different composers work as the highly diverse and personal nature of each composer's work practice makes comparisons difficult.
- *Gesture recognition is not mandatory*. Instead, gesture information is often treated as continuous streams of data to control the inputs of sound synthesis

algorithms[8] *without identifying discrete gestures*. In contrast, most gesture-based interactions in HCI—even in post-WIMP interfaces—make heavy use of gesture recognition to issue discrete commands. Crucially, the goals of the interactions are usually quite different (cf. Hunt and Kirk's attributes, above).
- As already mentioned, *practice and continuous learning are essential* in most musical interactions. This implies that whether qualitative or quantitative, *evaluations of computer music interactions should ideally be longitudinal,* i.e. done over a non-negligible time period (i.e. more than a few minutes), allowing for the development of expertise as well as the evolution of personal representations and musical concepts.
- Finally, *bi-manual input is the rule*. Most computer music systems will require the use of both hands (and perhaps other limbs, such as the feet or lips) as interactors. Hands may have complementary or mirror roles, depending on the type of interaction required.

1.2 The MIDWAY Music Interaction Workbench

Music composition and performance are extremely creative activities, which makes the design and evaluation of support tools a difficult task (Shneiderman 2009). In a survey of computer music interfaces, Pennycook argues that:

> Unlike text-editing environments, in which measures of productivity can be gathered empirically, in most musical settings, productivity and aesthetic value become hopelessly confused. ... A user interface that satisfies the needs of one musician in an efficient, well-ordered way may be awkward or even counterproductive for another musician. (1985)

For this reason, designing computer music systems that satisfy more than a single user or a single context of use can be problematic. However, several methods and tools have been proposed specifically to address these challenges.

A common approach is to provide designers and developers with a given technical framework along with design guidelines in which they will be able to experiment and realize new systems (Berthaut and Dahl 2015; Hödl et al. 2016). However, this approach often restricts the design choices to a subset of technical and aesthetic possibilities and does not generalize well to unexpected alternatives. Instead of creating a generic framework, we focus on the creation of a workbench. The goal is to provide useful theoretical and technical tools to support the creation and evaluation of musical interactions. Unlike a design framework or environment, the MIDWAY workbench contains conceptual models and evaluation techniques in addition to tools, avoids prescribing the kinds of instruments and systems that can be created, and can additionally form a point of reference around which different models and tools can be discussed. Finally, we include eight design guidelines adapted from the

---

[8] and/or synthesis for other modalities such as haptic or video displays.

literature and our own experience developing new DMIs and interactive music systems.

2 Models

HCI researchers have proposed many models to support the design and evaluation of interactive systems. Three are particularly relevant for musical interaction, with direct implications for our work:

2.1 Human Information Processing

Jens Rasmussen's *Human Information Processing* (Rasmussen 1986; Malloch et al. 2006) is a useful theoretical framework for making sense of the various interaction possibilities in music. It proposed three main interaction possibilities: *skill-level*, *rule-level* and *model-level* interactions, forming a continuum from embodied to mostly cognitive interaction (Fig. 1).

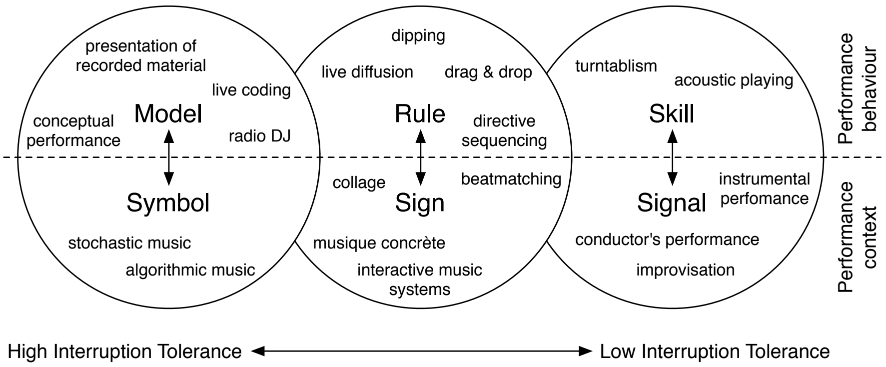

Fig. 1 Rasmussen's framework applied to musical interaction. The three performance behaviors are: skill-based, rule-based or model-based (top half). Performance contexts are shown on the bottom half. On the right, contexts demand close temporal coupling between performer and instrument, with little tolerance for interruption. On the left, contexts have much looser coupling. Adapted from (Malloch et al. 2006)

At *skill-level* in the musical domain, performers interact with their instruments in a very tight temporal relationship, allowing the potential for the performer's body schema to incorporate the instrument (Leman 2008) provided that a) the response of the instrument is sufficiently *modellable* by sensorimotor programs in the performer's nervous system[9], and b) the performer has accumulated sufficient experience and practice-time with the instrument. This is the case for both acoustic musical instruments and for digital musical instruments (DMIs) such as the T-Stick (Malloch and Wanderley 2007) or the Sponge (Marier 2017), which allow for "tight" interaction between the performer and the DMI using control that is continuous, integral (Jacob et al. 1994), and multiparametric (Hunt and Kirk 2000). At *rule-level*, performers interact with instruments/systems in a more detached way, by choosing from a set of already-learned actions. This applies to live-looping tools (Barbosa et al. 2017), with which performers build musical complexity by controlling only a subset of the available controls during each iteration of the loop, while previously-recorded processes are "performed" by the system. At *model-level*, performers consciously analyse a system before designing appropriate solutions. Therefore, from *skill-* to *model-level*, interaction between the user and the object with which they interact becomes more and more decoupled, implying less embodiment and more cognition, with the consequence that interaction also becomes more tolerant to interruptions. Composers frequently rely on this class of interaction to explore musical ideas through computational models (Garcia et al. 2014a).

Describing interaction with DMIs and interactive music systems using such a framework has important implications for instrument/system design. For instance, considering DMI mapping, designers might choose from:

1. *static* and *deterministic* mappings of DMIs versus the *dynamic* mappings of interactive music systems that may use *stochastic models*, *flocking behaviours*, or exhibit system *agency*, and
2. *complex* (*many-to-many*) mappings typical in advanced, skill-based DMIs versus *simpler one-to-one* mappings in *rule-based* or *model-based* interactions.

The framework also suggests how to design feedback for performers, including continuous signals for skill-based performance, signs for rule-based interaction, and symbols for model-based behavior. Consequently, Rasmussen's framework has implications for the design (i.e., presence and type) of haptic feedback for digital instruments and systems. For example, skill-based performances could benefit from continuous tactile and force-feedback to help enhance the interaction between player and instrument. On the other hand, rule- and model-based systems would be better served with tactile notifications based on short, discrete *tactons* (tactile icons) for user awareness instead of continuous signals tightly responding to user actions.

2.2 Instrumental Interaction

Beaudouin-Lafon's *Instrumental Interaction* (2000) describes how users interact with objects of interest, mediated by "Interaction Instruments" similar to interaction

[9] To be clear, we do not suggest the existence of an abstract system model in the performer's brain, but that sensorimotor programs can link the performer-instrument system in a way that affords *predictive control*.

with physical tools. In Instrumental Interaction, instruments are treated as *first-class objects* and can be defined in terms of *reification*, *polymorphism* and *reuse*. According to Beaudouin-Lafon and Mackay (2000):

- "*Reification* extends the notion of what constitutes an object", i.e. processes (e.g., commands) can be turned into first-class objects (including instruments) that can be manipulated directly by the user;
- "*Polymorphism* extends the power of commands with respect to these objects", i.e. commands (and instruments) can be applied to objects of different kinds;
- "*Reuse* provides a way of capturing and reusing patterns of use", these patterns can be previous user inputs and/or previous system outputs.

With its focus on the tool (instrument) and "object of interest" rather than system output, Instrumental Interaction seems like a good model for re-examining the design of interactive systems for music. Design of DMIs in the "traditional" category (continuous, multiparametric control and physically-embodied interfaces) already treat the instrument as a "first-class" object, especially when considered from the perspective of embodied interaction. However, other types of musical interaction may be ripe for reinterpretation through the lens of Instrumental Interaction.

The historical development of modular analog synthesis has left us with instrument/object-based models of signal processing that are now ubiquitous in modern digital music systems—for example, it is common to treat a "filter" as either a tool to be applied to static content or a process to link into a signal-processing chain. There remains a large number of musical concepts and constructs that are not treated this way, and could probably be *reified* into first-class objects, for example instrumental ensembles, rhythmic grids, harmonic systems, and sequences or scores.

With respect to *reuse*, in addition to usage patterns of the reified instruments mentioned above, patterns from actual performance with interactive systems (e.g., gestures, entire performances) could also be make available for reuse.

Our application of the Instrumental Interaction model to interactive music system design is still exploratory at this stage, and we hope to have concrete example applications to provide as part of the workbench documentation.

2.3 Co-adaptation

Co-adaptive systems (Mackay 2000) involve both *learning* by the user, who needs to understand what the system is capable of and how to access its functionality, and *appropriation* by the user, who needs to understand how to modify the system to meet his or her specific needs. In the first case, the user adapts to the system, as it is, and in the second case, the user adapts the system to meet future needs. Co-adaptation can take place at different timescales, from immediate real-time interaction to long-term interaction over time. Similarly, the scope of the interaction can vary, from the individual command level to large-scale, organized activities.

Musicians clearly co-adapt with their musical instruments (ibid.)—not only do they learn the constraints and possibilities of the instrument, but they also find ways to adapt or modify the instrument to achieve creative goals. Our goal is to create co-adaptive instruments that are explicitly designed for appropriation by musicians, using skill- and rule-based approaches together with the principles of polymorphism and reuse.

3 Tools

We describe two projects for which we designed modular and reusable tools that meet the idiosyncratic needs of musicians. These applications illustrate and support concepts defined in the above models, opening new possibilities for flexible musical applications.

3.1 LibMapper & ICon

Input Configurator (ICon) (Dragicevic and Fekete 2001) and libmapper (Malloch et al. 2014) are open-source software tools intended for the design and (re)configuration of modular interactive systems. Both focus on mapping and configuration as a top-level task separate from the task of designing input devices, target applications or media engines.

ICon is an interactive data-flow editor in which various input and interaction resources, such as input devices and communication protocols, can be instantiated as data-flow processing devices, and connected to each other and to interactive applications.

Libmapper was designed to support the creation of DMIs. Applications and input devices declare their local resources, which can be remotely discovered and connected over a local network. Various session-management tools can be used to interact with the resulting distributed network to add, modify or remove connections between producers and consumers of real-time data.

ICon and libmapper are complementary. ICon features a much richer visual programming interface with a large, extensible library of data processing devices, as well as both data-flow and state-machine approaches for describing and prototyping advanced interaction techniques (Appert et al. 2009). It follows the principles of Instrumental Interaction by reifying interaction techniques into data-flow processing devices and configurations that can be manipulated as first-class objects and applied to other contexts or systems (polymorphism and reuse). Libmapper has similar properties, but its distributed nature natively supports collaborative design, since an arbitrary number of users can interact with the same mapping network. It also adds support for supervised machine learning (ML) tools through the ability to stream or query the value of any signal in the network (including "destination" signals such as synth or application inputs) for providing training examples to the ML system. While recent versions of libmapper support “convergent” mapping topologies—in which multiple source signals are combined to control some destination parameter—representing and modifying the combining function can be confusing and problematic; ICon’s support for state-machine representations provides one possible solution.

For the workbench, we have built a prototype bridge between the two tools in order to exploit their complementarity: special ICon devices can give access to any libmapper signal present on the local network. Additionally, the UI enables the construction of "compound devices" that include signals from a variety of sources gathered together into a logical collection—effectively reifying the designer’s new instrument concept (Fig. 2). By leveraging the benefits of the two approaches, especially their support for the interaction models discussed above, we envision the design of musical applications that treat mapping configurations and interaction as first-class objects.

More recently, we have been exploring support tools for distributed versioning and collaborative annotation of mapping configurations as they evolve during development of a new DMI (Wang et al. 2017). These tools aim to provide support for comparing different mapping ‘vignettes’ developed during exploratory workshop sessions, and for discussing, recovering, and extending past mapping configurations. Lastly, we are considering approaches for evaluating the *compatibility* of mapping vignettes and if possible merging them into more complete, complex, and interesting instruments—a task that is greatly complicated by the different usage-patterns enabled and encouraged by different mapping configurations (Malloch and Wanderley 2017).

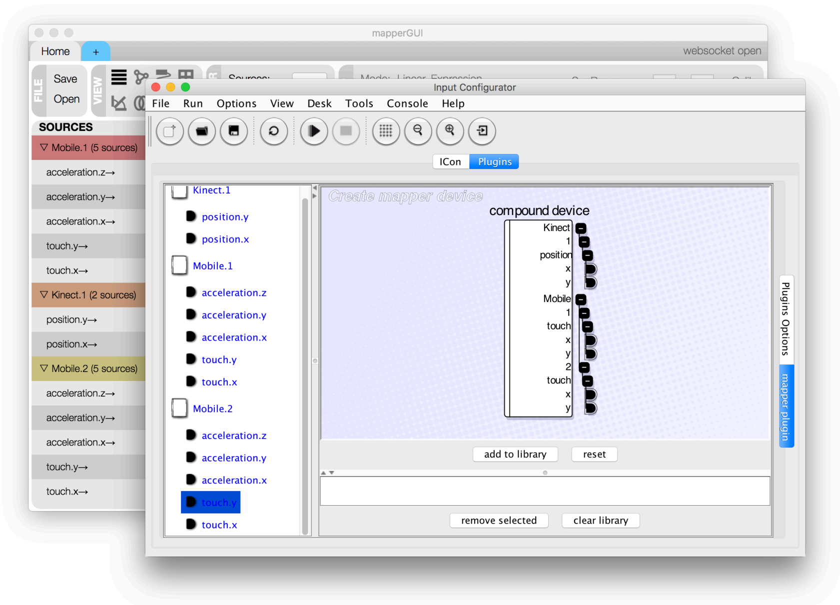

Fig. 2 A compound device is created in ICon using various signals coming from different physical devices and brokered by libmapper. This enables the construction of instrument models from high-level components; here a device consists of two mobile phones and a Kinect depth camera. Routing parts of the libmapper network through the bridge also allows the use of ICon's library of interaction models and the integration of complex state machines into the mapping configuration.

3.2 Paper Substrates and PaperComposer

Paper Substrates (Garcia et al. 2012) are physical paper components that support the creation of complex, user-defined interfaces for acting on musical data during the composition process. *PaperComposer* (Garcia et al. 2014b) provides a software "interface builder" that helps composers create, manage and then use their own interactive paper interfaces, which take advantage of Paper Substrates.

Instead of replacing current tools, Paper Substrates extend existing music programming environments by enabling composers to physically reify conceptual structures and create contexts for automatically interpreting handwritten input. Paper Substrates subscribe and/or publish to OSC data channels that act upon existing programming environments or other substrates. Individual paper components can be freely arranged, combined, and chained both spatially and in terms of their corresponding data channels; components are linked by drawing a stroke over overlapping papers using a digital pen. For example, a composer can combine a component that enters pitches via symbolic notation with another component that defines the pitches' amplitudes with a curve, producing a more complex, interactive instrument. This interaction technique is *polymorphic* since it applies to all Paper Substrates but the resulting data stream (or streams) will depend on the components' types. In this case we can consider the digital pen as an *interaction instrument* operating on Paper Substrates. In turn, the Paper Substrates also act as interaction instruments, operating on data or software objects residing in connected music programming environments.

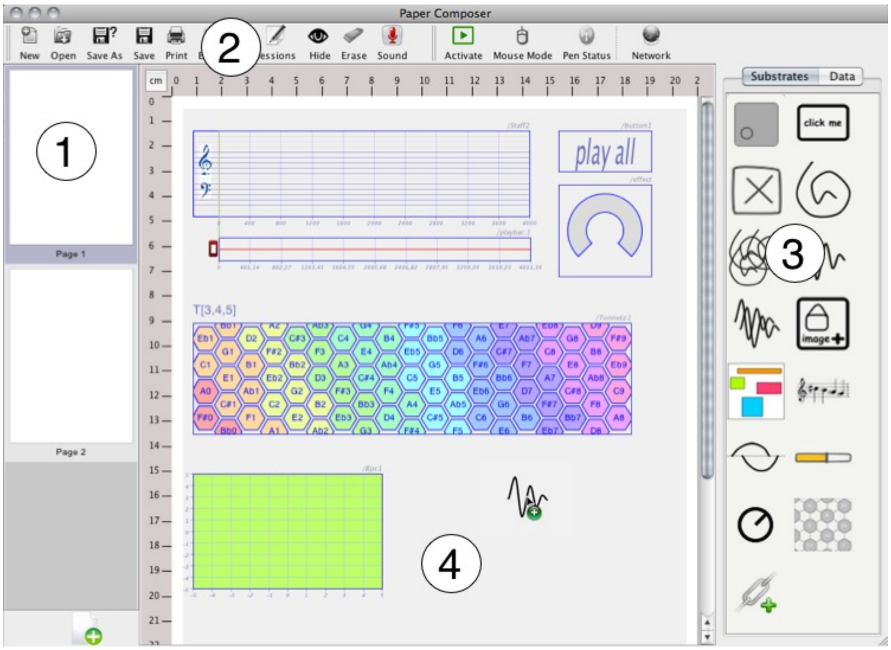

Fig. 3 PaperComposer interface: (1) Current document pages; (2) Toolbar to manage document and set parameters; (3) Thumbnails of available component classes that can be dragged into the virtual page to create new instances; and (4) Virtual page with components: musical staves, playbar, button, knob slider, tonnetz, curve container

Composers use the modular components of PaperComposer to create and customize their own paper interfaces for a range of musical creation tasks, such as drawing control curves or writing musical sequences with music notation (Fig. 4). New paper components can be developed and integrated within PaperComposer using a Java API, and then used and re-used by composers with any compatible application, supporting co-adaptation. In addition to support archiving and modification of paper documents, PaperComposer enables the user to store and reuse previously drawn pen strokes independently of the original Substrate.

Building upon PaperComposer and Paper Substrates, we designed *Polyphony*, a unified user interface that integrates interactive paper and electronic user interfaces, to study and compare the composition process of twelve expert composers from early sketches to final scores (Garcia et al. 2014c). Our challenge was to create conditions for comparing measures of qualitative and quantitative behavior while balancing control and external validity. We used a structured observation method with a composition task that was created and assessed with two composers: compose an electronic piece with an audio effect and a synthesizer, based on a recording of a 20-second musical piece by Anton Webern. Although this task is not representative of all real-world composition processes (it relies on a creative stimulus to shorten the ideation phase), it still requires key composition skills to produce an original musical result. All composers successfully completed the task and reported that they found the task challenging and fun. This methodology allowed us to obtain comparable snapshots of the composition process and reveal how composers both adapt and appropriate these instruments, each in their own way.

Fig. 4 Interacting on paper to explore computer-based musical processes. Photo H. Raguet © Inria

4 Design Guidelines

One of the key questions in the new-interface and computer music communities is how to approach the design of interfaces for musical expression. Given the wide availability of low-cost, user-friendly tools—sensors, actuators, microcontrollers, and a variety of software tools for creating/manipulating sound and defining mappings—everything seems possible. In practice, this situation often makes the designer's task even more difficult due to the bewildering breadth of choice and lack of technical constraints ("white page syndrome").

4.1 Underlying assumptions

We propose guidelines for guiding the design of interfaces for musical expression to be used in advanced musical contexts, i.e. requiring the development of performance practice or the representation and refinement of musical ideas over time. These guidelines are based on our experience studying and designing such interactive musical devices and tools.

We start by adopting four guidelines proposed by Hunt and Wanderley (2002), to which we add four new suggestions. The proposed guidelines are complementary to—though sometimes in conflict with—those proposed by Cook (2001; 2009). One of the reasons for the conflicts is an underlying assumption regarding the type of instrument being developed: those designed for immediate reward vs. interfaces meant for extensive musical practice and therefore requiring the development of expertise. Although it has been claimed that one could design DMIs for both types, i.e. combine "low entry fee with no ceiling in virtuosity" (Wessel and Wright 2002), both "entry" into musical practice and "virtuosity" within that practice will be defined by cultural context—contexts that may not yet exist for new instruments. In our experience it is better to focus on one group of performers (e.g. professional percussionists, or children visiting a museum) rather than attempting to design a universal instrument, especially considering that any "successful" DMI will be appropriated for use far outside of the designer's intent.

not commonly the case; interfaces and mapping that attract the interest and curiosity of professional musicians over the long-term tend to be difficult to learn and while we believe virtuosity is overemphasized in discussions of DMIs.

This difference in design intent does not imply any value judgement but has several crucial consequences, the most important of which is for the choice of mapping strategies: complex mappings typically tie variables together (Jacob et al.'s "integrality") (1994) and require learning or internalising more complex relationships while simple mappings tend to keep variables independent ("separability") and tend to offer more straightforward access to performing with such instruments. Note that the choice between complex or simple mappings is also related to the chosen sound generation algorithm.

4.2 Guidelines

The first four guidelines introduced by Hunt and Wanderley (2002) are:

1. ***Require energy for amplitude***
2. ***Two hands are better***
3. ***Use complex mappings—changes to one parameter should inflect others***
4. ***Control timbre in a non-direct manner***

In short, (1) sound amplitude is in direct proportion to the energy of the input gesture, (2) input devices allow performances using multiple limbs (e.g. two hands, but also possibly lips or feet) to provide more interaction options to the performer, (3) mappings that are not one-to-one typically require learning, but potentially provide more subtle control options and (4) as in most acoustic musical instruments, the control of timbral characteristics is not made using a dedicated control variable, but obtained by the combination of several variables.

The Sponge (Marier 2010; 2017) was designed following these guidelines. Marier and several other musicians have developed their own performance practice[10] using the Sponge (with a variety of abstract sound synthesis models in SuperCollider) and the instrument has been performed solo, in duets and with a laptop orchestra.

Based on our recent research, we now proposed four additional guidelines:

Guideline 5: Match integrality and separability of controls

Related to the third guideline, it is essential to consider the match between inputs and the task, in other words "the interrelationship between the perceptual structure of the task and the control properties of the device" (Jacob et al. 1994). This guideline separates control variables into two classes: *Integral* and *Separable*. In short, if a task is perceived as *integral*, then the structure of the controls should be integral and vice-versa.

As a simple example, imagine controlling a synthesis space consisting of three dimensions: if these dimensions are perceived as separable, for instance controlling the loudness of three recorded sounds, then three sliders (separable controls) would do the job—this is why mixing desks are widely used! On the other hand, if one wants to control the X, Y, and Z positions of a sound source in space, then an integral controller such as a 3DoF mouse would work well. Now think about inverting this situation and using three independent sliders to control the position of sound in space and the 3DoF mouse to independently change the loudness of each recorded sound. Both could actually work, but the interaction would be far from natural.

We have encountered this issue again and again in various designs over the last twenty years. We have found that these rules invariably apply, despite claims that they are not universal (Martinet 2012).

Guideline 6: Consider the speed of interaction when choosing inputs

Works by Wanderley et al. (2000) and by Marshall and colleagues (2006; 2009) explored an interesting question initially raised by Vertegaal et al. (1996): is there a particular match between transducers (i.e. sensors, in this case), musical function (static/dynamic; relative/absolute) and feedback modalities (visual/tactile/kinesthetic)? If so, this match could predict the type of sensor needed to control a certain musical feature (considering a one-to-one mapping). The problem with the initial work is that the proposed match was based on the authors' previous experience, not on experimental data. We carried out several experiments to confirm or refute the

[10] video showing how to approach the instrument: https://youtu.be/FMU8YAYiqos

proposed match, mostly focusing on the control of relative dynamic musical functions, such as a vibrato.

We found that, as proposed, the relationship between pressure sensors and relative dynamic functions seems to hold well. As predicted, an isometric pressure transducer (e.g. a force-sensing resistor) was generally preferred over other types of transducer (linear or rotary position). But the most important aspect in the choice of sensors was actually how fast the interaction was taking place (Marshall et al. 2009): up to around 3Hz, the sensors all yield similar results; for faster movements, some sensors seem more fitted to certain tasks. This sheds new light onto the design as it adds another independent variable (speed), which is not commonly taken into account.

Guideline 7: Support personal strategies and representations

Instead of forcing users to conform to an existing framework, tools need to support musicians creating their own conceptual representations. The tools presented in this chapter have been designed to support this kind of flexibility: for libmapper, by enabling compatibility between idiosyncratic, user-defined systems; and for Paper-Substrates, by supporting free arrangement and remixing of a variety of graphical data representations.

Guideline 8: Use multiple parallel representations

Music composition and performance usually involve different phases of creation, from ideation to score production or from rehearsals to actual performances. For each of these phases, musicians rely on different representations and tools each with specific advantages for the task at hand. e.g. sketching on paper, assessing a sound synthesis algorithm with real time feedback or practicing electronic pieces with a reduced tempo.

Instead of proposing a single environment to support the whole creative process, we found that user interfaces that integrate several input and output modalities can help creative practitioners to work effectively by using the most appropriate modality for the task at hand during the whole composition process. For instance, the *Polyphony* interface synchronized a rich set of existing music composition tools, including pen-based input with either interactive paper or a graphics tablet, as well as a keyboard, mouse, and physical controllers. Composers especially appreciated Polyphony's ability to synchronize across input devices and the live feedback it offers in a common workspace (Garcia et al. 2014c). Designers should consider crafting tools that are interoperable so that users can create ad hoc compound systems that take advantage of multiple input and output modalities in parallel.

5 Conclusion and Future Work

In this chapter, we argue for the creation of a "musical interaction workbench" that brings together models and tools from HCI and computer music in order to support the creation of new interactive technologies for musical composition and performance. The tools introduced above are examples of technologies that embed knowledge, in that they have been explicitly designed to support design principles and conceptual models such as Instrumental Interaction. They have been used by performers, composers and instrument designers, have supported the creation of numerous public performances around the world, and have been used to study the composition process.

In addition to the collection of models and tools that constitute the workbench, we formulated and discussed practical design guidelines that we hope will benefit other musicians, composers, designers and researchers.

We are now working on the next generation of our design workbench, which will include more concrete and practical design guidelines within the tools themselves. For example, libmapper could suggest signal connections based on well-known HCI theories and models such as integrality and separability of input devices or bimanual interaction. ICon could provide visual feedback on the properties and "quality" of mappings based on similar theories and models, and support more advanced visual programming tools, e.g. for specifying how a Paper Substrate should interpret pen input.

References

1. Appert C, Huot S, Dragicevic P, Beaudouin-Lafon M (2009) FlowStates : prototypage d'applications interactives avec des flots de données et des machines à états. In: Proceedings of ACM/IHM 2009, p 119–128
2. Barbosa J, Wanderley MM, Huot S (2017) Exploring playfulness in NIME design: the case of live looping tools. In: Proceedings of the 2017 international conference on new interfaces for musical expression, Copenhagen, Denmark, p 87–92
3. Beaudouin-Lafon M (2000) Instrumental Interaction: An interaction model for designing post-WIMP user interfaces. In: Proceedings ACM CHI, p 446-453
4. Beaudouin-Lafon M, Mackay WE (2000) Reification, polymorphism and reuse: three principles for designing visual interfaces. In: Proceedings of ACM/AVI, p 102–109
5. Berthaut F, Dahl L (2015) BOEUF: A unified framework for modeling and designing digital orchestras. In: International symposium on computer music multidisciplinary research, Springer International Publishing, p 153–166
6. Cook P (2001) Principles for designing computer music controllers. In: ACM-CHI NIME workshop, Seattle, USA. Reprinted and expanded with author's and expert's comments in Jensenius AR, Lyons MJ (eds) A NIME reader: fifteen years of new interfaces for musical expression, Springer International Publishing, 2017, p 1–13
7. Cook P (2009) Re-designing principles for computer music controllers: a case study of SqueezeVox Maggie. In: Proceedings of the international conference on new interfaces for musical expression, Pittsburgh, USA, p 218–221
8. Dragicevic P, Fekete JD (2001) Input device selection and interaction configuration with ICON. In: Blandford A, Vanderdonckt J, Gray P (eds) Proceedings of IHM-HCI 2001. People and Computers XV - Interaction without frontiers. Lille, France. Springer Verlag, p 543–448
9. Eaglestone B, Ford N (2001) Composition systems requirements for creativity: what research methodology? In: Proceedings of the MOSART workshop, p 7–16

10. Fiebrink R, Trueman D, Britt C, Nagai M, Kaczmarek K, Early M, Daniel MR, Hege A, Cook P (2010) Toward understanding human-computer interaction in composing the instrument. In: Proceedings of the international computer music conference, New York, USA
11. Garcia J, Tsandilas T, Agon C, Mackay WE (2012) Interactive paper substrates to support musical creation. In: Proceedings of ACM/CHI, p 1825–1828
12. Garcia J, Leroux P, Bresson J (2014) pOM: Linking pen gestures to computer-aided composition processes. In: Proceedings of the 40th international computer music conference joint with the 11th sound & music computing conference
13. Garcia J, Tsandilas T, Agon C, Mackay WE (2014) PaperComposer: creating interactive paper interfaces for music composition. In: Proceedings of ACM/IHM, Lille, France, p 1–8
14. Garcia J, Tsandilas T, Agon C, Mackay WE (2014) Structured observation with polyphony: a multifaceted tool for studying music composition. In: Proceedings of the 2014 conference on designing interactive systems, p 199-208
15. Hödl O, Kayali F, Fitzpatrick G, Holland S (2016) LiveMAP design cards for technology-mediated audience participation in live music. In: Mudd T, Holland S, Wilkie K (eds) Proceedings of the music and HCI workshop, CHI 2016, San Jose, USA
16. Huot S (2013) ‘Designeering interaction’: a missing link in the evolution of human-computer interaction. Habilitation à Diriger des Recherches, Université Paris-Sud XI, France
17. Hunt A, Kirk R (2000) Mapping strategies for musical performance. In: Wanderley MM, Battier M (eds) Trends in gestural control of music, Ircam Centre Pompidou, France, p 231–258
18. Hunt A, Wanderley MM (2002) Mapping performer parameters to synthesis engines. Organised Sound, 7(2):97–108
19. Jacob RJK, Sibert LE, McFarlane DC, Mullen MP (1994) Integrality and separability of input devices. ACM Transactions on Computer-Human Interaction, 1(1):3–26
20. Leman M (2008) Embodied music cognition and mediation technology, MIT Press
21. Mackay WE (2000) Responding to cognitive overload: Co-adaptation between users and technology. Intellectica, 30(1):177–193
22. Malloch J, Birnbaum D, Sinyor E, Wanderley MM (2006) Towards a new conceptual framework for digital musical instruments. In: Proceedings of the international conference on digital audio effects, Montreal, Canada, p 49–52
23. Malloch J, Wanderley MM (2007) The T-Stick: from musical interface to musical instrument. In: Proceedings of the international conference on new interfaces for musical expression, New York, USA, p 66–69
24. Malloch J, Sinclair S, Wanderley MM (2014) Distributed tools for interactive design of heterogeneous signal networks. Multimedia Tools and Applications, 74(15):5683–5707
25. Malloch J, Wanderley MM (2017) Embodied cognition and digital musical instruments: design and performance. In: Lesaffre M, Maes PJ, Leman M (eds) The Routledge companion to embodied music interaction, p 440–449
26. Marier M (2010) The Sponge: a flexible interface. In: Proceedings of the international conference on new interfaces for musical expression, Sydney, Australia, p 356–359
27. Marier M (2017) Musique pour éponges : La composition pour un nouvel instrument de musique numérique. D.Mus. Thesis, Université de Montréal, Canada
28. Marshall MT, Wanderley MM (2006) Evaluation of sensors as input devices for computer music interfaces. In: Kronland-Martinet R, Voinier T, Ystad S (eds) CMMR 2005 - Proceedings of computer music modeling and retrieval 2005 Conference. Lecture notes in computer science, vol 3902. Springer-Verlag, Berlin Heidelberg, p 130-139
29. Marshall MT, Hartshorn M, Wanderley MM, Levitin DJ (2009) Sensor choice for parameter modulations in digital musical instruments: empirical evidence from pitch modulation. Journal of New Music Research, 38(3):241–253
30. Martinet A, Casiez G, Grisoni L (2012) Integrality and separability of multi-touch interaction techniques in 3D manipulation tasks. IEEE Transactions on Visualization and Computer Graphics, 18(3):369–380

31. Medeiros CB, Wanderley MM (2014) A comprehensive review of sensors and instrumentation methods used in musical expression. Sensors Journal, 14(8):13556–13591
32. Miranda ER, Wanderley MM (2006) New digital musical instruments: control and interaction beyond the keyboard. A-R Editions, Madison, USA
33. Pennycook BW (1985) Computer-music interfaces: a survey. ACM Computing Surveys, 17(2):267-289
34. Rasmussen J (1986) Information processing and human-machine interaction: an approach to cognitive engineering. Elsevier Science Inc, New York, USA
35. Shneiderman B (2009) Creativity support tools: a grand challenge for HCI researchers. In: Engineering the User Interface, Springer, London, p 1–9
36. Vertegaal R, Ungvary T, Kieslinger M (1996) Towards a musician's cockpit: transducers, feedback and musical function. In: Proceedings of the international computer music conference, Hong Kong, China, p 308–311
37. Wanderley MM, Viollet JP, Isart F, Rodet X (2000) On the choice of transducer technologies for specific musical functions. In: Proceedings of the international computer music conference, Berlin, Germany, p 244–247
38. Wanderley MM, Orio N (2002) Evaluation of input devices for musical expression: borrowing tools from HCI. Computer Music Journal, 26(3):62–76
39. Wang J, Malloch J, Huot S, Chevalier F, Wanderley MM (2017) Versioning and annotation support for collaborative mapping design. In: Proceedings of the sound and music computing conference, Espoo, Finland
40. Wessel D, Wright M (2002) Problems and prospects for intimate musical control of computers. Computer Music Journal, 26(3):11–22